PRODUCT REQUIREMENTS DEVELOPMENT AND MANAGEMENT PROCEDURE

PREPARED BY:
PAT SCHULER AND TOM SHULL
NASA LANGLEY RESEARCH CENTER

NOTE: THIS DOCUMENT WAS PREPARED BY AND FOR THE LANGLEY RESEARCH CENTER FOR REVIEW AND VOLUNTARY ADOPTION BY PROJECTS CONDUCTED AT THE CENTER. THE GOAL IS SUBSEQUENT ESTABLISHMENT AS A MANAGEMENT SYSTEM PROCEDURE. IT IS BASED ON "BEST PRACTICES" AS TAUGHT IN THE NASA NET PROGRAM SYSTEM REQUIRMENTS (REQ) COURSE. THE PRIMARY REFERENCE IS THE TEXT "CUSTOMER CENTERED PRODUCTS: CREATING SUCCESSFUL PRODUCTS THROUGH SMART REQUIREMENTS MANAGEMENT", I. F. HOOKS AND K. A. FARRY, AMACOM (2001), ISBN 0-8144-0568-1. APPLICATION FEEDBACK IS WELCOME AND ANY QUESTIONS OR COMMENTS SHOULD BE DIRECTED TO THE SECOND AUTHOR, WHO MAY BE REACHED VIA EMAIL AT THOMAS.A.SHULL@NASA.GOV

PRODUCT REQUIREMENTS DEVELOPMENT AND MANAGEMENT

Objectives:
- To define the process to generate, document, review, baseline, and manage requirements for a system of interest
- To ensure customer and user requirements are understood and met
- To define a set of complete, consistent, unambiguous, and verifiable requirements

Approval ________________________________
TBD

Table of Contents

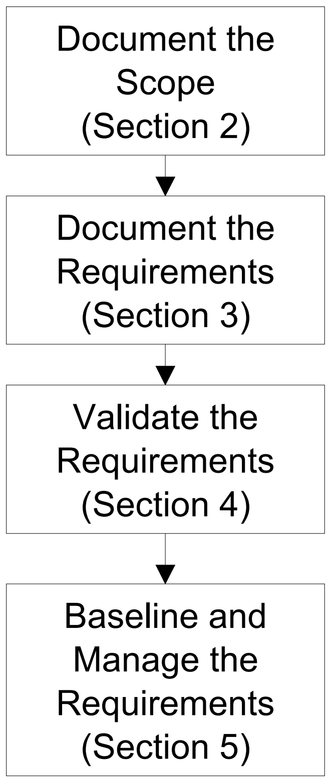
Section 1: Overview of Procedure

1.1 How to use this document

- This procedure applies to the higher-level systems of interest, but not to the lowest levels, such as the mechanical part level or software subroutine. See the project's System Engineering Management Plan or equivalent for the applicable architectural levels to which this procedure applies.
- This procedure applies to the development of all types of product requirements (e.g. System Requirements, Software Requirements, Interface Requirements, etc.). The term Requirements Document is used generically throughout this procedure.
- Refer to the project's Document Templates for instructions on capturing individual items described below and organizing Requirements Documents.
- Use the project's Glossary to ensure consistent use of terminology.
- The context diagram below (Section 1.2) defines, at a high level, the activities of this procedure. Sections 2 through 5 expand the activities shown in the context diagram into flow diagrams and detailed steps.
- Some of the boxes in the flow diagrams are numbered with their associated step, which provides detailed instructions for performing the actions in that box. If no number is given, there are no additional details provided.
- Apply the Section 2 through 5 flowcharts recursively to each system of interest (e.g., system, flight segment, ground segment, instrument, spacecraft).

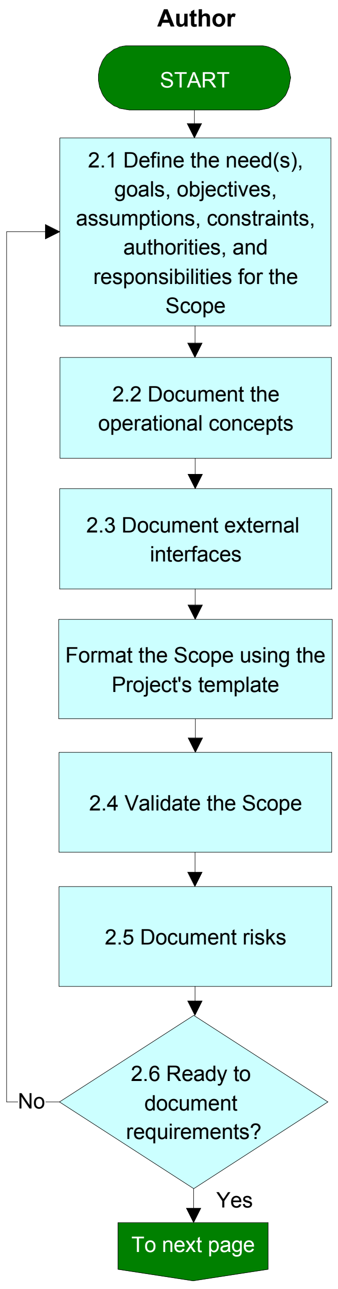
1.2 Context Diagram

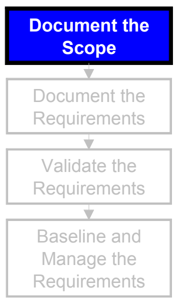
This Product Requirements Development and Management Procedure is made up of four major phases as shown below.


Document the Scope (Section 2)
Document the Requirements (Section 3)
Validate the Requirements (Section 4)
Baseline and Manage the Requirements (Section 5)

Section 2: Document the Scope (of the system of interest)

Instructions: The actions defined in the boxes below are performed in parallel and iteratively.


Author
START
2.1 Define the need(s), goals, objectives, assumptions, constraints, authorities, and responsibilities for the Scope
2.2 Document the operational concepts
2.3 Document external interfaces
Format the Scope using the Project's template
2.4 Validate the Scope
2.5 Document risks
2.6 Ready to document requirements?
No
Yes
To next page


Document the Scope
Document the Requirements
Validate the Requirements
Baseline and Manage the Requirements

Inputs/Entry Criteria:

- All stakeholder groups are identified and the names of the representatives are documented.
- Parent system of interest documents are available i.e. parent Scope, Requirements, IRDs, and Interface Control Documents (ICDs).
- The authors have been trained on how to develop and document the scope.

Outputs/Exit Criteria:

- Scope documentation is reviewed and approved for a feasible approach that will satisfy the product Need.
- Risks have been identified and mitigation plans developed per the project's Risk Management Plan.

STEP	ACTION TO BE TAKEN
2.1	**Define the need(s), goals, objectives, assumptions, constraints, authorities and responsibilities for the Scope** a. Become familiar with parent system-of-interest documents. b. Document the need(s). Get all stakeholders aligned to one vision of the need through discussions and reviews. GUIDANCE: The 'need' is not a definition of the product or solution. The 'need' explains why the project is developing this product from the stakeholder's point of view (i.e. What problem the stakeholders want to solve). The 'need' is not likely to change much during the project. Each stakeholder has an individual perspective, agenda, priorities, constraints, understanding of the environment, alternate solutions, and lessons learned. A common set of expectations / vision / need among stakeholders must be formed by sharing knowledge about these perspectives, ensuring that the expectations are realistic, and obtaining agreement by all stakeholders on the needs. Push for a single need statement, but recognize there are exceptions to this rule. If more than one need is identified, ask if they could be stated as a goal or objective All stakeholders should agree to a common vision of what the 'need' is before the scope definition can be completed and a full set of quality requirements can be generated. c. Document the goals that define what has to be accomplished to meet the need(s). 1. Elicit and document goals from all stakeholders. 2. Analyze the goals, ensure alignment with need, obtain missing information, and resolve conflicts. 3. Prioritize the goals. d. Document the objectives that define, in measurable terms, how the project plans to meet the goals. GUIDANCE: Objectives are initiatives that implement the goals. The objectives also specify the success criteria (i.e. What is the minimum that the stakeholders expect from the system for it to be successful?). e. Explicitly document all constraints (i.e., external items that cannot be controlled and that must be met, which are identified while defining the scope). f. Document the specific authority and responsibility (e.g., to contractor, project technical lead, customer) for aspects of the products development, identified while defining the scope. g. Explicitly, document all assumptions that are identified while defining the need(s), goals, objectives, constraints, authority, and responsibility. h. Validate assumptions (e.g., obtain confirmation from stakeholders, perform prototype or study to determine feasibility).
2.2	**Document the operational concepts** a. Document operational concepts and associated scenarios for each operational mode, mission phase (e.g., installation, startup, typical examples of normal and contingency operations, shutdown, and maintenance), and lifecycle phase (e.g., development, integration, test, and validation) from all stakeholders' points of view. GUIDANCE: Operational scenarios are a step-by-step description of how the proposed system should operate and interact with its users and its external interfaces (e.g. other systems). Imagine the operation of the future product and document, from the stakeholder's perspective, the steps of how the end-to-end system will function or be used. Choose scenarios that best fit the needs. For additional guidance on operational concept and scenarios, see IEEE 1362 [5]. 1. Document 'Nominal' scenarios (i.e. scenarios that cover 'normal' operations and environments). Consider the questions: Who will use the product? Why? Where? When? How? Under what conditions? Use the interface diagram discussed in Step 2.3 while discussing and developing the operational concepts with the stakeholders. 2. Document 'Off-Nominal' scenarios (i.e. scenarios to cover abnormal operations and environments). Consider the following: hazards to users, others, the product, other products; potential misuse of the product; extreme conditions. 3. Refine the scenarios to cover all interfaces. Cover the following interactions: inputs expected; outputs expected; input does not occur; wrong input occurs; wrong output occurs. b. Validate the scenarios by iterating on the above until all the stakeholders agree on the correctness, completeness, and feasibility of the scenarios.
2.3	**Document external interfaces** a. Document all external interfaces, including those to enabling systems, which are identified while documenting the need, goals, objectives, and operational concept. Document all the potential interfaces as well (e.g. those for power, structural or physical, data or signal, command or control).

GUIDANCE: The external interfaces form the boundaries between the system of interest and the rest of the world.

b. Ask the following questions about each boundary to the system of interest, considering each product lifecycle phase (development through operation and disposal) and document the answer:
   - What does the product do to/for the world?
   - What does the world do to/for the product?
   - What is the worst thing that can happen across this interface?
   - Is the interface likely to change during the development of the product?
   - Is this interface likely to change after the product is in use?

c. Create the external interface diagram to the system of interest. The diagram should depict all the interfaces documented in the steps above.

d. Document every industry standard, application programmer's interface, or ICD that exist for the external interfaces.

e. Identify IRDs that must be developed for each external interface that does not exist or does not have an ICD.

f. Document internal interfaces information, as it is determined.
   GUIDANCE: Internal interfaces are not addressed in detail at this point. Because dividing a system of interest into lower level systems of interest is a design task, the author should leave a hole in the Requirements Document for these internal interface requirements. The author can then go back and fill this hole when far enough into design to know where these interfaces should be; this means the author may have to submit a change request. An alternative to leaving a hole in the Requirements Document is to develop a separate IRD for each interface; this adds other documents with associated authority and controls.

g. Use the following questions to verify interfaces.
   - Have you identified and documented all product interfaces?
   - Have you located ICDs for interfaces to existing products?
   - Have you created a mechanism to monitor interface changes outside your control?
   - Have you involved people from the other side of the external interface in the interface definition effort?
   - Have you simplified interfaces as much as possible?
   - Have you distributed the product interface documentation?
   - Have you created a mechanism for tracking interfaces through development to ensure that reality matches the documentation?

h. Maintain the interface diagram(s).

2.4 **Validate the Scope**

Conduct reviews with representatives from all stakeholder groups to:
- determine validity of the scope,
- ensure alignment with parent system of interest documents,
- identify problems and risks,
- find helpful suggestions,
- make sure everyone has the same expectations of the system of interest,
- ensure scope represents a feasible approach to meeting need, and
- approve scope.

GUIDANCE: Once the Scope seems to be fairly stable, consider conducting an inspection of the documented Scope to eliminate remaining errors, following the Instructional Handbook for Formal Inspection [6].

2.5 **Document risks**

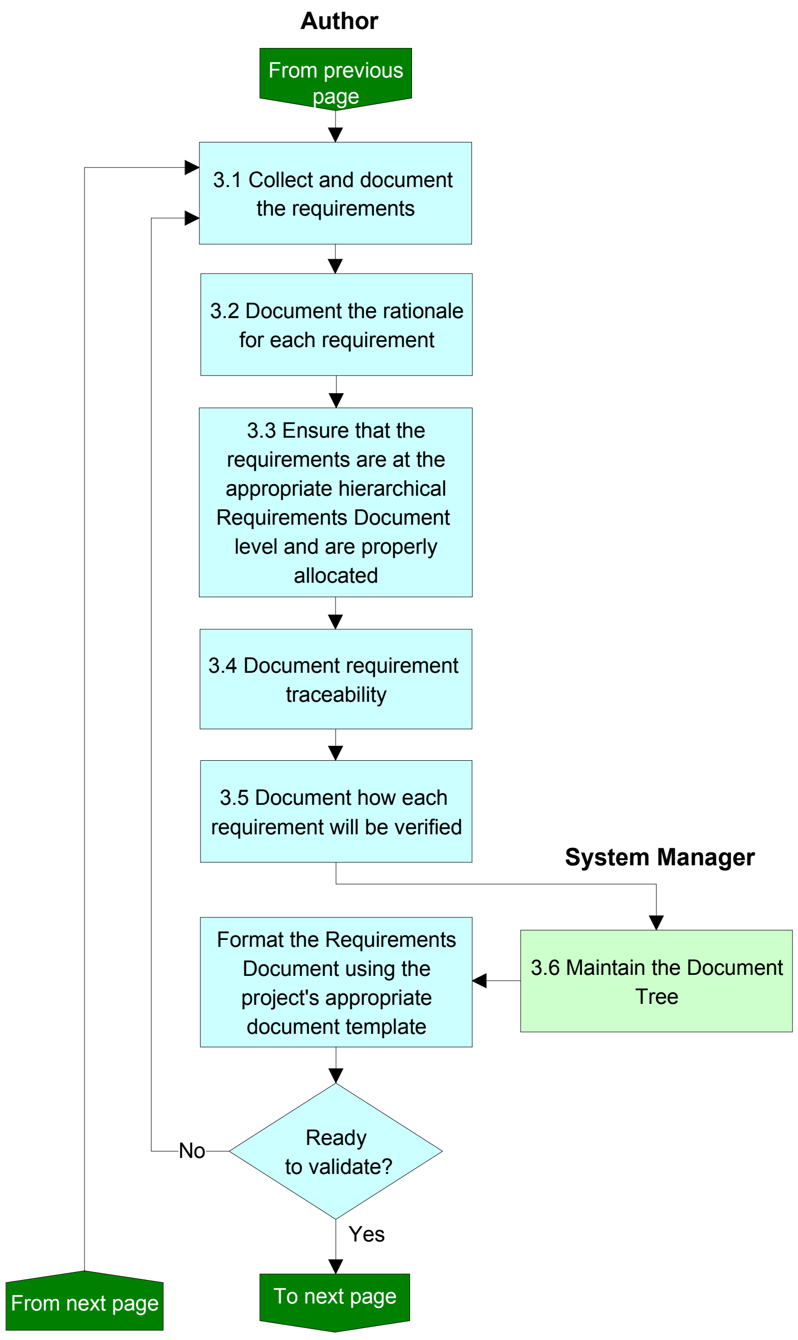
Determine and document risks associated with the recorded Scope following the project's Risk Management Plan. Use the following questions to help identify risks. *A"YES" answer indicates risk.*
- Do we have product boundary questions?
- Have we missed a key stakeholder input?
- Have we missed a product lifecycle phase?
- Are there poorly defined or incomplete interfaces?
- Are there areas of strong disagreement?
- Are there too many unknowns?
- Are there assumptions that have not been confirmed with the project or stakeholder personnel?
- Are there technical issues?

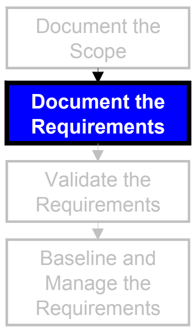
	• Are there technology issues? • Are there schedule issues (e.g., overly optimistic)? • Are there cost issues (e.g., budget too lean)? • Are there too many uncertainties?
2.6	**Ready to document requirements?** Decide if ready to proceed to documenting the requirements. **If** / **Then** High risk: The requirements writing phase should be postponed until the risk can be mitigated. Repeat the appropriate steps above to reduce the risk. Low risk: Put mitigation plan into action and start requirements writing. If the scope is rolled out as a separate document from the requirements, gain approval and baseline the scope document before proceeding.

If	Then
High risk	The requirements writing phase should be postponed until the risk can be mitigated. Repeat the appropriate steps above to reduce the risk.
Low risk	Put mitigation plan into action and start requirements writing. If the scope is rolled out as a separate document from the requirements, gain approval and baseline the scope document before proceeding.

Section 3: Document the Requirements (of the system of interest)

Instructions: The actions defined in the boxes below are performed in parallel and iteratively. When the flow has been completed for portions of the Requirements Document, those portions can be moved on to Section 4 of this procedure.


Author
From previous page
3.1 Collect and document the requirements
3.2 Document the rationale for each requirement
3.3 Ensure that the requirements are at the appropriate hierarchical Requirements Document level and are properly allocated
3.4 Document requirement traceability
3.5 Document how each requirement will be verified
System Manager
3.6 Maintain the Document Tree
Format the Requirements Document using the project's appropriate document template
Ready to validate?
No
Yes
From next page
To next page


Document the Scope
Document the Requirements
Validate the Requirements
Baseline and Manage the Requirements

Inputs/Entry Criteria:

- The reviewed and approved Scope for this system of interest.
- The authors have been trained on how to develop and document good requirements.

Outputs/Exit Criteria:

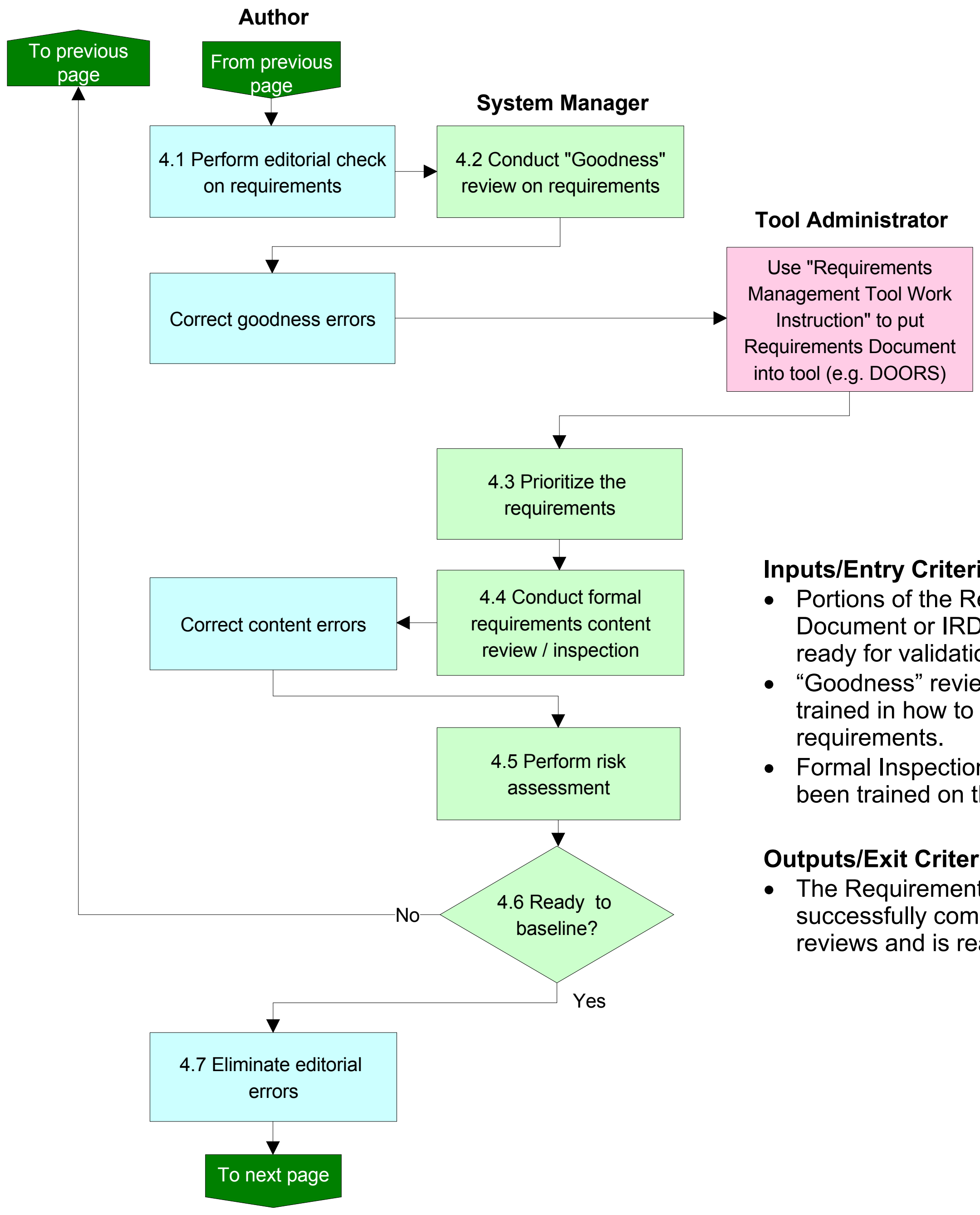
- Portions of the Requirements Document or IRD are completed and ready for validation.

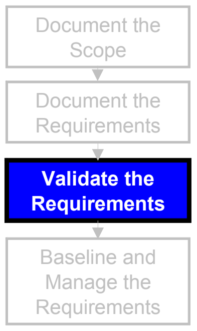
STEP	ACTION TO BE TAKEN
3.1	**Collect and document the requirements** a. Become familiar with system of interest parent documents and the Scope for this system of interest. b. With the scope of this system of interest as a starting point, and with input from the stakeholders, collect and document each individual requirement for this system of interest following the Rules for Writing Good Requirements (A.1 - A.4). GUIDANCE: Requirements Documents should only contain product requirements. Project requirements are generally captured in a Statement of Work (SOW) or project plans.
3.2	**Document the rationale for each requirement** As each requirement is documented, record (per the project's template) the rationale which includes of the following items: • The reason for the requirement (i.e., why requirement exists and the source of the requirement) GUIDANCE: Often the reason for the requirement is not obvious and it may be lost if not recorded as the requirement is being documented. The reason may point to a constraint, trade or design study, or operations concept. If there is a "traceability link" from a higher-level requirement that completely explains the reason for the requirement, then simply reference the link. • Assumptions made while developing the requirement. Assumptions must be confirmed before the requirements can be baselined. • The relationships with the product's expected operations (e.g. expectations about how customers will use a product) GUIDANCE: This may be done with a link to the Operational Concept. • High-level design choices that drive low-level requirements (e.g. trade study results) GUIDANCE: If the requirement states a method of implementation, the rationale must state why the solution is being limited to this one method of implementation.
3.3	**Ensure that the requirements are at the appropriate hierarchical Requirements Document level and are properly allocated** a. Choose the correct architectural level, i.e., system of interest, for documenting requirements. Use the following questions to aid in determining if the requirements are at the appropriate Requirements Document level. If the answer to any of these questions is no, make the appropriate correction. • Does "why do we need the requirement" take you back directly to the level above? • Does the requirement allow you more then one architecture or design option for the next level? • Does it make sense to verify the requirement at this level? • Have all constraints that apply to this level been captured? b. After preliminary decomposition of the system of interest into lower level systems of interest, document (per the project's template) the allocation of each requirement at this level to the next lower level system of interest that must accomplish the requirement. This allocation is necessary so that the author at the next level knows exactly which parent-level requirements apply. c. Use the questions below to help verify the correctness of the requirements allocations. • Is every requirement allocated? • Are there any duplicate or conflicting requirements from parent source Requirements Documents? • Can the system of interest to which you have allocated the requirement do it alone? Or should it be allocated to more than one system of interest? • Are interfaces simple and controllable? (If not, this may indicate a potential architectural problem.)
3.4	**Document requirements traceability** a. As each requirement is documented, record (per the project's template) its lineage/traceability to at least one parent at the next higher level. b. Use the following checklist to validate the documented traceability. (It is preferable to have someone other then the author perform this activity.) • Trace each requirement back to requirements (or scope, for the top-level requirements) in the level above it and vise versa. The requirement should be evaluated to assure that the requirements trace is correct and that it fully answers the parent requirements. If it does not, some other requirement(s) must complete fulfillment of the parent requirement. • If there is no parent, is the requirement "gold plating" or is there a missing requirement at the higher level?

	• Resolve duplication between levels. If a requirement is simply repeated at a lower level and it is not an externally imposed constraint, perhaps the requirement does not belong at the higher level. c. The author corrects all traceability errors.
3.5	**Document how each requirement will be verified** a. For each requirement, document the verification method(s) (i.e. inspection, demonstration, analysis, or test), the level, and the phase. If a requirement is unverifiable, it must be rewritten. GUIDANCE: Levels of verification are, for example, component, subsystem, and system. Phases of verification are, for example, design, manufacture, verification. b. Document any new / additional requirements that are uncovered during determination of the verification method (e.g., extra connectors on the wiring harness to connect to test instrumentation or external power, extra data on a display or in a database to give visibility into an internal process).
3.6	**Maintain the Document Tree** Use the following checklist to aid in developing and maintaining the Document Tree. • Segregate requirements into documents of manageable size and along organizational management lines. • Segregate requirements that may be revised frequently into separate documents. • Segregate requirements that will be contractually binding to each outside party, and create separate Requirements Documents for each party. • Identify approval levels for the separate Requirements Documents.

Section 4: Validate the Requirements (of the system of interest)

Instructions: The actions defined in the boxes below are performed sequentially.


Author
To previous page
From previous page
System Manager
4.1 Perform editorial check on requirements
4.2 Conduct "Goodness" review on requirements
Tool Administrator
Correct goodness errors
Use "Requirements Management Tool Work Instruction" to put Requirements Document into tool (e.g. DOORS)
4.3 Prioritize the requirements
Correct content errors
4.4 Conduct formal requirements content review / inspection
4.5 Perform risk assessment
No
4.6 Ready to baseline?
Yes
4.7 Eliminate editorial errors
To next page


Document the Scope
Document the Requirements
Validate the Requirements
Baseline and Manage the Requirements

Inputs/Entry Criteria:
- Portions of the Requirements Document or IRD are completed and ready for validation.
- “Goodness” reviewers have been trained in how to document good requirements.
- Formal Inspections participants have been trained on the method.

Outputs/Exit Criteria:
- The Requirements Document has successfully completed all the reviews and is ready to be baselined.

STEP	ACTION TO BE TAKEN
4.1	**Perform editorial check on requirements** a. Perform an editorial check using the Editorial Checklist in A.2. GUIDANCE: Consider having someone other than the author perform this activity. Use "redlines" so the author can check to see if the change would corrupt the meaning. b. The author corrects all editorial errors.
4.2	**Conduct "Goodness" review on requirements** a. Conduct a "Goodness" review using the Goodness Checklist in A.3. This review is conducted with 2-3 trained technical staff. b. Provide redlines/review comments back to the author.
4.3	**Prioritize the requirements** a. Determine when to prioritize the requirements. Prioritization must be performed before design. GUIDANCE: Prioritization can be performed now (i.e., to take advantage of input on the priorities from the stakeholders during the upcoming review) or performed after the review (i.e., deferred until a firmer set of requirements is established). Prioritize before design to guide architecting, design tradeoffs, system "builds", and prototyping; enable effective downstream descopes due to schedule and resource shortfalls; and manage requirement additions and risk. b. Define the schema that will be used to prioritize the requirements. GUIDANCE: A numbering scheme of 1-2-3 may be used, where 1 = essential, mandatory, nonnegotiable (i.e. minimum to satisfy customer need, not susceptible to trades), or urgent requirements; 2 = useful (enhance the product value to the customer, these could be written as a "should"), negotiable (subject to trade), or slightly deferrable requirements and 3 = desirable if low cost, can be readily descoped, flexible, or longer delay requirements. c. With input from all the stakeholders (including the developers), document each requirement's priority. GUIDANCE: Ask all stakeholders to classify the requirements by priority. Use the operational scenarios to help in the classification. Often, it is easiest to identify the 1's and 3's first and allow everything else to default to 2. d. Resolve the priority differences between stakeholders. e. Maintain the priorities throughout development. GUIDANCE: Prioritization is not finished until the last version of the product is done. Priorities are most likely to change when there are major budget or schedule changes; when new information exists about cost, schedule, or technical feasibility; when external interfaces change; or when the designers want a priority change to match design. Also, when the customer brings new requirements that require deferring some existing requirement, reassess priorities to make certain that the least important priorities are the ones being deferred.
4.4	**Conduct formal requirements content review / inspection** a. Determine whether to perform a formal review or Formal Inspection. GUIDANCE: Formal reviews differ from inspection in that (1) formal reviews frequently cover the entire document while inspections cover only a limited number of pages per inspection; (2) inspections identify defects (inspectors may provide suggestions outside of the inspection meeting) while formal reviews can require participants to both identify defects and provide recommended corrections; and (3) defects are dispositioned as part of the inspection meeting while formal reviews disposition both the defects and the corrections. b. Include representatives from all stakeholders in the review / inspection. Also, consider inviting outside experts for their project or domain experience. GUIDANCE: Reviewers experienced in different lifecycle phases and aspects of use will be best equipped to find the omissions and incorrect facts in those phases and aspects. Include reviewers from developers, maintainers, manufacturers, testers, installers, and especially users. c. Educate your review team on what constitutes a good requirement. GUIDANCE: Assign the reading of Chapter 7, pages 101-119 [3] to reviewers who have not already had training on writing good requirements. d. Perform the review. **If** / **Then** Formal review / Follow Formal Review Instructions in B.1. Formal Inspection / Follow Formal Inspection Instructions in B.2. e. Ensure the entire Requirements Document for this system of interest has successfully undergone Formal Review or Formal Inspections prior to performing the next activity in this procedure.

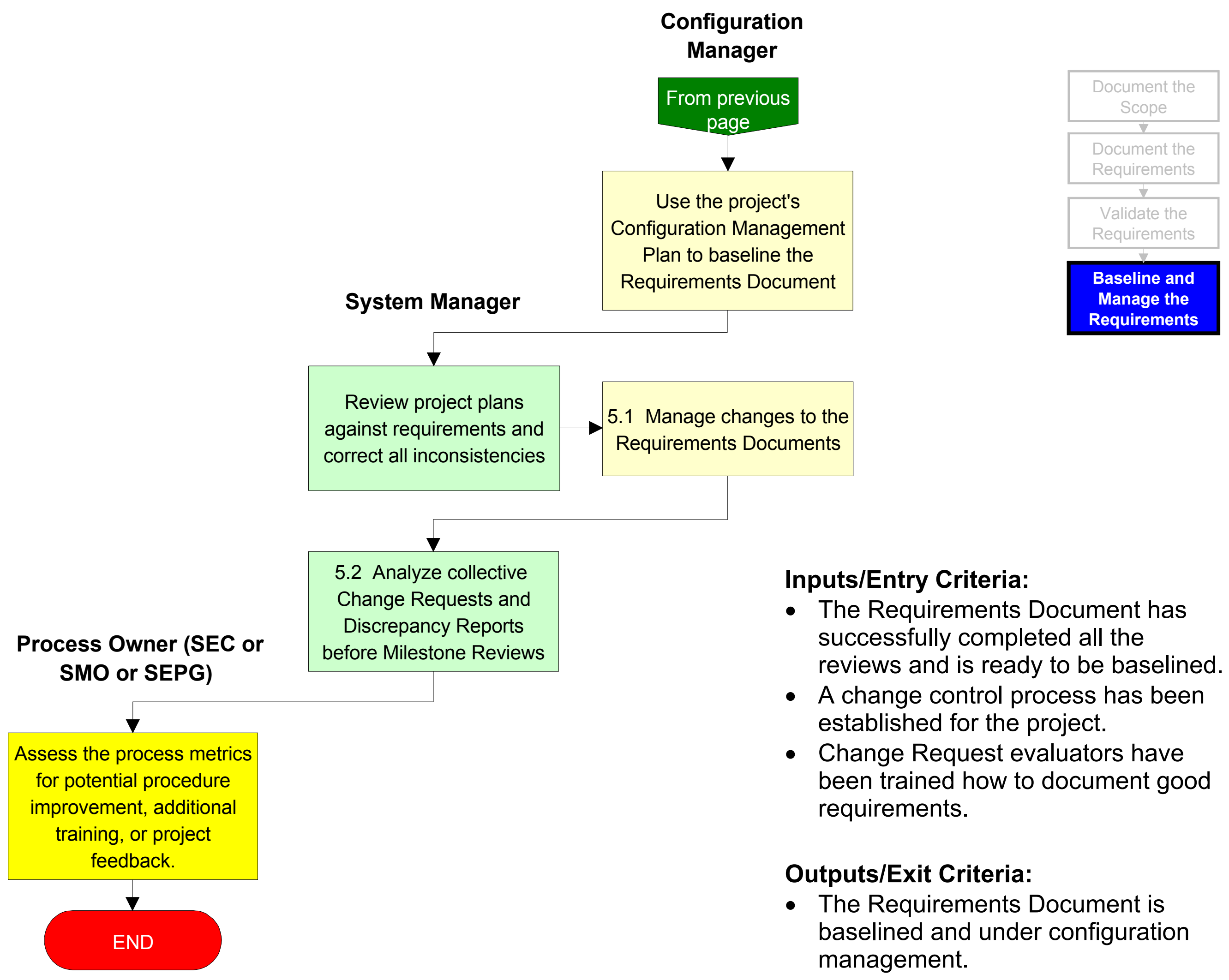
4.5	**Perform risk assessment**. a. Address requirements volatility risks: 1. Identify volatile requirements. 2. Modify sensitive requirements to eliminate the need to change the requirement if the volatility becomes reality or add additional requirements to make the design robust to volatility. 3. Develop a plan to manage the development effort through requirement volatility when eliminating sensitivity is not possible. GUIDANCE: If the requirements are volatile (i.e., likely to change) consider assigning a stability rating to each requirement. For example: use a scheme of ‘high-medium-low’ where high = it is highly likely that the requirement will change; medium = the requirements may change; and low = the requirement is very unlikely to change. b. Identify technical feasibility risks of each requirement and where possible, modify requirements to reduce the risk and stay consistent with budget and schedule. c. Ensure that the schedule and budget are realistic for the set of requirements to be included in the baseline. GUIDANCE: Refer to page 193-198 [3] for additional information on how to reduce requirements risks.
4.6	**Ready to baseline?** Decide if ready to baseline the Requirements Document. **If** / **Then** High risk: Postpone the Requirements Document baselining until the risk can be mitigated (i.e., repeat the appropriate steps above to reduce the risk). Low risk: Proceed to the next step to prepare the Requirements Document for baselining.
4.7	**Eliminate editorial errors** a. Use the Editorial Checklist in A.2 to ensure that each requirement is editorially correct. (Consider having someone other then the author perform this activity. Use “redlines” so the author can check to see if the change would corrupt the meaning.) GUIDANCE: As requirement defects discovered during the review / inspection are corrected, editorial problems may inadvertently result. A final check before baselining can reduce formal changes later. b. The author corrects all editorial errors. c. Submit the Requirements Document for baselining.

Nested table in 4.6:

If	Then
High risk	Postpone the Requirements Document baselining until the risk can be mitigated (i.e., repeat the appropriate steps above to reduce the risk).
Low risk	Proceed to the next step to prepare the Requirements Document for baselining.

Section 5: Baseline and Manage Requirements (of the system of interest)

Instructions: The actions defined in the boxes below are performed throughout the project.


Configuration Manager
From previous page
Use the project's Configuration Management Plan to baseline the Requirements Document
System Manager
Review project plans against requirements and correct all inconsistencies
5.1 Manage changes to the Requirements Documents
5.2 Analyze collective Change Requests and Discrepancy Reports before Milestone Reviews
Process Owner (SEC or SMO or SEPG)
Assess the process metrics for potential procedure improvement, additional training, or project feedback.
END
Document the Scope
Document the Requirements
Validate the Requirements
Baseline and Manage the Requirements

Inputs/Entry Criteria:
- The Requirements Document has successfully completed all the reviews and is ready to be baselined.
- A change control process has been established for the project.
- Change Request evaluators have been trained how to document good requirements.

Outputs/Exit Criteria:
- The Requirements Document is baselined and under configuration management.
- Process improvement metrics have been collected and analyzed.

STEP	ACTION TO BE TAKEN
5.1	**Manage changes to the Requirements Documents** a. Using the project's Change Request (CR) form, document the requested requirements statement(s) change with justification. Note: All requirements must include a rationale, allocation, traceability, and verification method. b. Evaluate new or modified requirements against the checklists in A.1 – A.3 and applicable items in A.4. Work with the submitter to correct any deficiencies identified. c. Perform a thorough change impact assessment from the standpoint of all relevant stakeholders. Include impacts on parent and child requirements, existing commitments, cost, and schedule. Communicate potential changes to all who are impacted. d. Using the project's Configuration Management process to approve or disapprove each change. e. For approved changes: • Distribute the changes to all who are impacted • Maintain the requirements change history • Make approved change to the Requirements Document • Ensure all project plans, and other affected work products are updated to reflect the changes
5.2	**Analyze collective Change Requests and Discrepancy Reports before Milestone Reviews [9]** a. At the end of each development phase (e.g., preliminary design, detailed design, low-level testing, system-level qualification/acceptance testing), review CRs to identify nontrivial requirement changes (e.g. changes that affected design, required rework, impacted commitments, cost, or schedule). Sort those changes into three categories: • Modified • Added • Deleted GUIDANCE: For space flight projects, changes that effect schedule, cost, or technical performance are referred to as Class I [10]. b. Analyze these CRs to determine the cause of the changes and if improvements can be done to eliminate further changes. Provide feedback to this procedure's author if the analysis indicates changes/improvements in this procedure are warranted. c. At the end of each development phase, review product Discrepancy Reports (DRs) to identify those that involve requirements problems (i.e. separate problems in meeting requirements from requirements problems). Sort DRs involving requirements problems into three categories: • Incorrect requirement • Ambiguous or misunderstood requirement • Missing requirement GUIDANCE: For space flight projects, DRs are referred to as Nonconformance-Failure Reports (NFRs) [10]. d. Record the following metrics: • The name of the project and the system of interest • Phase currently completing • Date • The total number of requirements • The number of requirement changes in each of the three categories (Modified, Added, Deleted) • The total number of DRs involving requirements in each of the three categories (Incorrect, Ambiguous, Missing) GUIDANCE: The Requirements Metrics Collection Worksheet (see [11] for sample) can be used to record the metrics. e. Present at milestone reviews (e.g., PDR, CDR, TRR, and SAR) the percent of requirements changes in each category, the results of the CR analysis, and the percent of discrepancies related to requirements problems in each category. GUIDANCE: This information will provide the reviewers with an indication of the stability and quality of the requirements, the associated risks, and readiness to proceed to the next phase.

Appendix A: Rules for Writing Good Requirements

A.1 Use Correct Terms

Shall = requirement; **Will** = facts or declaration of purpose; and **Should** = goal.

A.2 Editorial Checklist

- The requirement is in the form "product ABC shall XYZ". A requirement must state "The product shall (do, perform, provide, weigh, or other verb followed by a description of what), i.e., must be in "**Who**" shall "**What**" form; uses active rather than passive voice.
  Example Product requirements:
    - The system shall operate at a power level of…
    - The software shall acquire data from the…
    - The structure shall withstand loads of…
    - The hardware shall have a mass of…
- The requirement uses consistent terminology to refer to the product and its lower level entities.
- The requirement is grammatically correct.
- The requirement is free of typos, misspellings, and punctuation errors.
- The requirement complies with the project's template and style rules.

A.3 Goodness Checklist

Is each requirement…

- Clear and understandable?
  - Can only be understood one way?
  - Free from indefinite pronouns (this, these)
  - Expressing only one thought per requirement statement? A standalone statement (as opposed to multiple requirements in a single statement or a paragraph that contains both requirements and rationale)?
  - Stated simply and concisely?
  - Stated positively (as opposed to negatively, e.g. "shall not")?
- Free of ambiguities (e.g., as appropriate, etc., and/or, support, but not limited to, be able to, be capable of)?
- Free of unverifiable terms (e.g., flexible, easy, sufficient, safe, ad hoc, adequate, accommodate, user-friendly, useable, when required, if required, appropriate, fast, portable, light-weight, small, large, maximize, minimize, sufficient, robust, quickly, easily, clearly, other "ly" words, other "ize" words)?
- Free of implementation? (Requirements should state WHAT is needed, NOT HOW to provide it, i.e. state the problem not the solution. Ask, "Why do you need the requirement?" The answer may point to the real requirement.)
- Free of descriptions of operations? (Don't mix operation with requirements; update the Operational Concept instead. To distinguish between operations and requirements ask the questions: "Does the developer have control over this? Is this a need the product must satisfy or an activity involving the product?" Sentences like "The operator shall…" are almost always operational statements not requirements.)
- Free of TBDs (a best guess, marked TBR, with the rationale should replace these)?
- Complete with tolerances for qualitative/performance values (e.g., Less than, greater than or equal to, plus or minus, 3 sigma root sum squares)?
- Accompanied by intelligible rationale, including any assumptions? Can you validate (Do I concur with) the assumptions? Assumptions must be confirmed before baselining.
- Traceable to the next higher level (e.g., need, goals, objectives, constraints, operational concepts, interface requirements or parent requirements)?
- Identified with a verification method(s) (i.e., test, demonstration, analysis, or inspection)? Does a means exist to measure its accomplishment? Can you state the criteria required for verification? Can compliance be verified?
- Located in the proper section of the document?
- Defined at the correct level?
- Unique (as opposed to redundant)?
- Consistent with other requirements (as opposed to conflicting)?

A.4 Content Review / Inspection Checklist

CLARITY
1. Are the requirements clear and unambiguous? (i.e., are there aspects of the requirement that are not understood; can the requirement be misinterpreted?)
2. Are the requirements concise and simple?

COMPLETENESS
1. Are requirements stated as completely as possible? Have all incomplete requirements been captured as TBRs?
2. Are any requirements missing? For example have any of the following requirements areas been overlooked: functional, performance, interface, environment (development, manufacturing, test, transport, storage, operations), facility (manufacturing, test, storage, operations), transportation (among areas for manufacturing, assembling, delivery points, within storage facilities, loading), training, personnel, operability, safety, security, appearance and physical characteristics, and design.
3. Have all assumptions been explicitly stated?

COMPLIANCE
1. Are all requirements at the correct level (i.e. system, segment, element, subsystem etc.)?
2. Are requirements specified in an implementation-free way so as not to obscure the original requirements, i.e. do the requirements state "what" and not "how"?
3. Are requirements specified in an operations-free way? Is this a requirement the developer has control over, something the product must do, or a quality it must have, rather than an activity involving the product?

CONSISTENCY
1. Are the requirements stated consistently without contradicting themselves or the requirements of related systems?
2. Is the terminology consistent with the user and sponsor's terminology? With the project glossary?
3. Is the terminology consistently used through out the document?
4. Are the key terms included in the project's glossary?

TRACEABILITY
1. Are all requirements needed? Is each requirement necessary to meet the parent requirement? Is each requirement a needed function or characteristic? Distinguish between needs and wants. If it is not necessary, it is not a requirement. Ask, "What is the worst that could happen if the requirement was not included?"
2. Are all requirements (functions, structures, and constraints) traced to mission or system of interest scope (i.e. need(s), goals, objectives, constraints, or operational concept)?
3. Is each requirement stated in such a manner that it can be uniquely referenced in subordinate documents?
4. Is allocation to the next lower level documented?

CORRECTNESS
1. Is each requirement correct?
2. Is each stated assumption correct? Assumptions must be confirmed before the document can be baselined.
3. Are the requirements technically feasible?

FUNCTIONALITY
1. Are all described functions necessary and together sufficient to meet mission and system goals and objectives?

PERFORMANCE
1. Are all required performance specifications and margins listed? (e.g., consider timing, throughput, storage size, latency, accuracy and precision).
2. Is each performance requirement realistic?
3. Are the tolerances overly tight? Are the tolerances defendable and cost-effective? Ask, "What is the worst thing that could happen if the tolerance was doubled or tripled?"

INTERFACES
1. Are all external interfaces clearly defined?
2. Are all internal interfaces clearly defined?
3. Are all interfaces necessary, sufficient, and consistent with each other?

MAINTAINABILITY
1. Have the requirements for system maintainability been specified in a measurable, verifiable manner?
2. Are requirements written to be as weakly coupled as possible so that ripple effects from changes are minimized?

RELIABILITY
1. Are clearly defined, measurable, and verifiable reliability requirements specified?
2. Are there error detection, reporting, handling, and recovery requirements?
3. Are undesired events (e.g., single event upset, data loss or scrambling, operator error) considered and their required responses specified?
4. Have assumptions about the intended sequence of functions been stated? Are these sequences required?
5. Do these requirements adequately address the survivability after a software or hardware fault of the system from the point of view of hardware, software, operations personnel and procedures?

VERIFIABILITY / TESTABILITY
1. Can the system be tested, demonstrated, inspected, or analyzed to show that it satisfies requirements?
2. Are the requirements stated precisely to facilitate specification of system test success criteria and requirements?

DATA USAGE
1. Where applicable, are “don’t care” conditions truly “don’t care”? (“Don’t care” values identify cases when the value of a condition or flag is irrelevant, even though the value may be important for other cases.) Are “don’t care” conditions values explicitly stated? (Correct identification of “don’t care” values may improve a design’s portability.)

Appendix B: Formal Review / Inspection Instructions

B.1 Formal Review Instructions

1. Prepare the review package, which includes the document to be reviewed, background materials, instructions for the review, and Review Item Disposition (RID) form (see [11] for sample). Instructions include the following: a) read the entire package before writing any RIDs (including the background material such as scope and operational concepts), b) use Content Review / Inspection Checklist in A.4 to aid in identifying errors, c) document comments and corrections on the provided RID form, d) return completed RIDs in electronic form.
   GUIDANCE: At a minimum the RID form should require: (a) the original requirement (unless missing), (b) description of problem, (c) recommended new requirement or requirement / document change, and (d) justification (why the change is needed).
2. Conduct an overview meeting with the review team. At this meeting, review the scope and operational concepts of the system of interest. Provide the reviewers with the review package and brief them on the contents, the type of comments you want, the schedule, and method of submitting the RIDs.
3. Obtain completed RIDs.
4. Disposition the RIDs.
   a. Sort the recommendations into three groups: (1) definitely accept (or accept with modifications) (2) maybe accept; and (3) definitely not accept. Use the following checklist to help determine if a recommendation should be accepted.
      Acceptance Criteria Checklist:
      - Is the recommended addition within scope?
      - Is the recommended addition necessary to meet the scope?
      - Does the recommended change correct an error or assumption?
      - Does the recommended change address technical feasibility or constraints?
      - Does the recommended change clarify a requirement that can be misinterpreted?
      - Is the recommended deletion out of scope, gold-plating, or duplicate?
   b. For the second group, conduct meetings with the reviewers, authors, and key stakeholders to resolve issues; and, for the third group, document the reasons for disapproval.
   c. For those RIDs that have been accepted, evaluate new or modified requirements against the checklists in A.1 – A.3 and applicable items in A.4. Work with the submitter to correct any deficiencies identified.
   d. Incorporate acceptable reviewer recommendations.
5. Communicate and Revise the Disposition
   a. Circulate to all review participants a review recommendation summary showing the disposition of all RIDs.
   b. Hold a meeting with the reviewers to provide them an opportunity to comment on the recommendation summary.
6. Revise the requirements based on the outcome of the meeting.

B.2 Formal Inspection Instructions

Conduct the Formal Inspection following the instructions defined in the Instructional Handbook for Formal Inspections [6]. In addition, add the following to the planning stage activities of the handbook:

a. Include completion of the following three activities to the Formal Inspection entrance criteria:
   - Perform terminology check (A.1)
   - Perform editorial check (A.2)
   - Perform goodness check (A.3)
b. Include Content Review / Inspection Checklist in A.4 in the inspection package that is distributed.

Appendix C: Acronyms, Definitions, and References

C.1 Acronyms

CDR – Critical Design Review
CR – Change Request
DR – Discrepancy Report
ICD – Interface Control document
IRD – Interface Requirements Document
PDR – Preliminary Design Review
RID – Review Item Disposition
SAR – System Acceptance Review
SOW – Statement of Work
TBD – To be determined
TBR – To be resolved
TRR – Test Readiness Review

C.2 Definitions

Constraints: external items the project cannot control that must be met (e.g. regulations, work on existing systems, the product is needed by a certain date). During requirements development, constraints evolve into requirements or are used as the rational for the requirements. [8, p. 28]

Interface Control Document: defines the interface for existing products. [3, p.87]

Interface Requirements Document: defines the interface requirements for products that do not exist yet and must be developed. [3, p.87]

Change Request (CR): generic term referring the to the artifact used to submit and disposition requested changes to baseline documents.

(Critical) Milestone Reviews: the life-cycle series of rigorous system-level technical and programmatic evaluations conducted at key formulation and implementation milestones. Key milestones in this context are the major transition points in the life cycle, such as the transition from requirements development to design activities, final design to manufacturing, and the transition from the assembly and integration of components to system-level environmental testing. May include, but are not limited to, System Concept Review, Requirements Review, Preliminary Design Review, Critical Design Review, Pre-Environmental Review, Pre-Ship Review, and Operational Readiness Review. [9, p.120]

Discrepancy Report (DR): generic term referring the to the artifact used to officially report discrepancies or nonconformance resulting from verification and validation of products or systems.

Phases of the Lifecycle: formulation, design, development, manufacturing, verification, validation, shipping, storage, installation, pre-deployment, deployment, training, operations, maintenance, upgrading, disposal.

Product Requirement: a single statement of something the product must do or a quality the product must have. [2]

Project Requirement: a single statement of a task to be done or what the provider will deliver. [2]

Requirements Document: See definition for Requirements Specification. Note: The term Requirements Document and Requirements Specification are used synonymously.

Requirements Specification: A document that specifies the requirements for a system *(of interest)* or component. Typically included are functional requirements, performance requirements, interface requirement, design requirements, and development standards. [4]

Scope (of the system of interest): the need(s), goals and objectives, constraints (including budget and schedule), operational concepts, external interfaces, and associated assumptions.

Stakeholders: anyone who has a vested interest in the project (i.e.. systems engineering, electrical engineering, mechanical engineering, software engineering, manufacturing, testing, handlers of packaging, storing, shipping, and disposal, trainers, users during training, operations, and upgrades, operations, maintenance, and logistics).[8, p. 28] "A group or individual that is affected by or in some way accountable for the outcome of an undertaking. Stakeholders may include project members, suppliers, customers, end users, and others." [1]

System of Interest: The entity for which the engineering team is responsible for designing, developing, integrating, and testing. [7]

Validation (of requirements): The process of confirming the completeness and correctness of the requirements [3, p.157]. Validation is performed after requirements or changes to them are defined.

Verification (of system of interest): Verification is a process (consisting of tests, inspections, demonstrations, and analysis) of confirming that the designed and built product meets the requirements. [3]

C.3 References

1 CMMI for Systems Engineering/Software Engineering/Integrated Product and Process Development/Supplier Sourcing, Staged Representation; Version 1.1, Software Engineering Institute (2002), Technical Report CMU/SEI-2002-TR-012.

2 Conducting a Requirements Review, Class, Reference Card, Feb. 2003.

3 Customer Centered Products: Creating Successful Products Through Smart Requirements Management, I. F. Hooks and K. A. Farry, AMACOM (2001), ISBN 0-8144-0568-1.

4 IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology.

5 IEEE Standard 1362-1998, Guide for Information Technology System Definition Concept of Operation.

6 Instructional Handbook for Formal Inspections. (URL: http://sw-eng.larc.nasa.gov/process/)

7 Systems Engineering NPG Draft as of March 2003.

8 Systems Requirements class slides, given at LaRC Oct. 2002.

9 NPG 7120.5B, NASA Program and Project Management Processes and Requirements

10 LAPG 5300.1, Space Product Assurance.

11 Requirements Development and Management Web Site, URL: http://sw-eng.larc.nasa.gov/docs/requirements_capture_and_management.html